Task_4_prodigy_infotech_internship

June 4, 2024

PRODIGY INFOTECH DATA SCIENCE INTERN

#TASK 4

TASK OVERVIEW: Analyze and visualize sentiment patterns in social media data to understand public opinion and attitudes towards specific topics or brands.

IMPORTING THE LIBRARIES AND DATASET

[ ]:
```
import pandas as pd
from textblob import TextBlob
import re
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer
```

[ ]:
```
# Load the dataset
df = pd.read_csv("/content/twitter datset.csv",encoding="latin-1")
```

[ ]:
```
df.columns = ['Header1', 'company','labels','text']
```

[ ]:
```
df.head()
```

[ ]:
```
   Header1      company    labels  \
0     2401  Borderlands  Positive
1     2401  Borderlands  Positive
2     2401  Borderlands  Positive
3     2401  Borderlands  Positive
4     2401  Borderlands  Positive

                                                text
0  I am coming to the borders and I will kill you…
1  im getting on borderlands and i will kill you …
2  im coming on borderlands and i will murder you…
3  im getting on borderlands 2 and i will murder …
4  im getting into borderlands and i can murder y…
```

[ ]:
```
import nltk
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data…
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data…
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to /root/nltk_data…
```

[ ]:
```
True
```

DATA PREPROCESSING

[ ]:
```
def preprocess_text(text):
    # Check if text is not NaN
    if isinstance(text, str):
        # Convert text to lowercase
        text = text.lower()

        # Remove special characters, URLs, mentions using regex
        text = re.sub(r'http\S+|www\S+|@[^\s]+', '', text)
        text = re.sub(r'\W', ' ', text)
        text = re.sub(r'\s+', ' ', text)

        # Tokenize the text
        tokens = word_tokenize(text)

        # Remove stopwords
        stop_words = set(stopwords.words('english'))
        filtered_tokens = [word for word in tokens if word not in stop_words]

        # Lemmatization
        lemmatizer = WordNetLemmatizer()
        lemmatized_tokens = [lemmatizer.lemmatize(word) for word in␣
 ↪filtered_tokens]

        # Return the preprocessed text
        return ' '.join(lemmatized_tokens)
    else:
        return ''  # Return an empty string for NaN values
```

[ ]:
```
df['clean_text'] = df['text'].apply(preprocess_text)
```

# Sentiment Analysis using TextBlob

[ ]:
```
# Sentiment Analysis using TextBlob
def analyze_sentiment(text):
    analysis = TextBlob(text)
    return analysis.sentiment.polarity

df['sentiment_score'] = df['clean_text'].apply(analyze_sentiment)
```

[ ]:
```
df
```

[ ]:
```
       Header1      company    labels  \
0         2401  Borderlands  Positive
1         2401  Borderlands  Positive
2         2401  Borderlands  Positive
3         2401  Borderlands  Positive
4         2401  Borderlands  Positive
…          …            …         …
74676     9200       Nvidia  Positive
74677     9200       Nvidia  Positive
74678     9200       Nvidia  Positive
74679     9200       Nvidia  Positive
74680     9200       Nvidia  Positive

                                                    text  \
0      I am coming to the borders and I will kill you…
1      im getting on borderlands and i will kill you …
2      im coming on borderlands and i will murder you…
3      im getting on borderlands 2 and i will murder …
4      im getting into borderlands and i can murder y…
…                                                    …
74676  Just realized that the Windows partition of my…
74677  Just realized that my Mac window partition is …
74678  Just realized the windows partition of my Mac …
74679  Just realized between the windows partition of…
74680  Just like the windows partition of my Mac is l…

                                              clean_text  sentiment_score
0                                     coming border kill              0.0
1                               im getting borderland kill              0.0
2                              im coming borderland murder              0.0
3                            im getting borderland 2 murder              0.0
4                             im getting borderland murder              0.0
…                                                    …              …
74676  realized window partition mac like 6 year behi…             -0.4
74677  realized mac window partition 6 year behind nv…             -0.4
74678  realized window partition mac 6 year behind nv…             -0.4
74679  realized window partition mac like 6 year behi…             -0.5
74680  like window partition mac like 6 year behind d…             -0.4
```

```
[74681 rows x 6 columns]
```

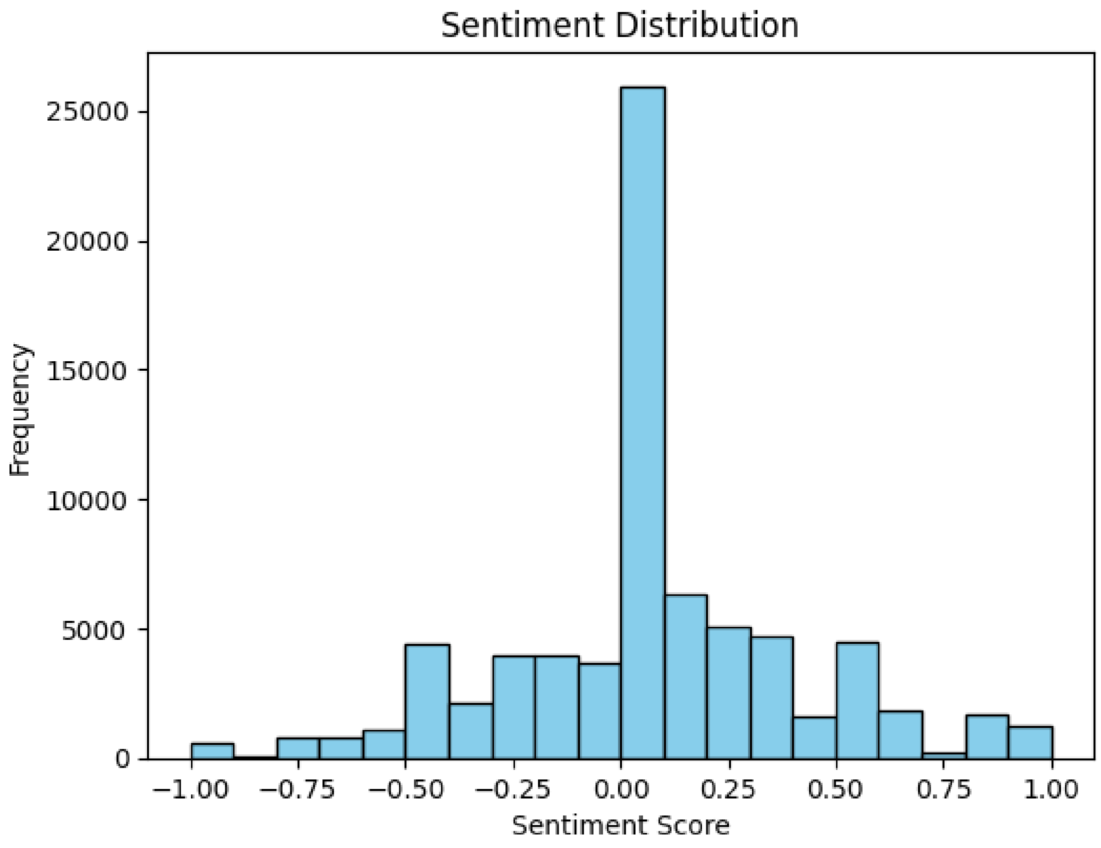
```
Warning: total number of rows (74681) exceeds max_rows (20000). Limiting to
first (20000) rows.
```

##VISUALIZAION

Plot sentiment distribution

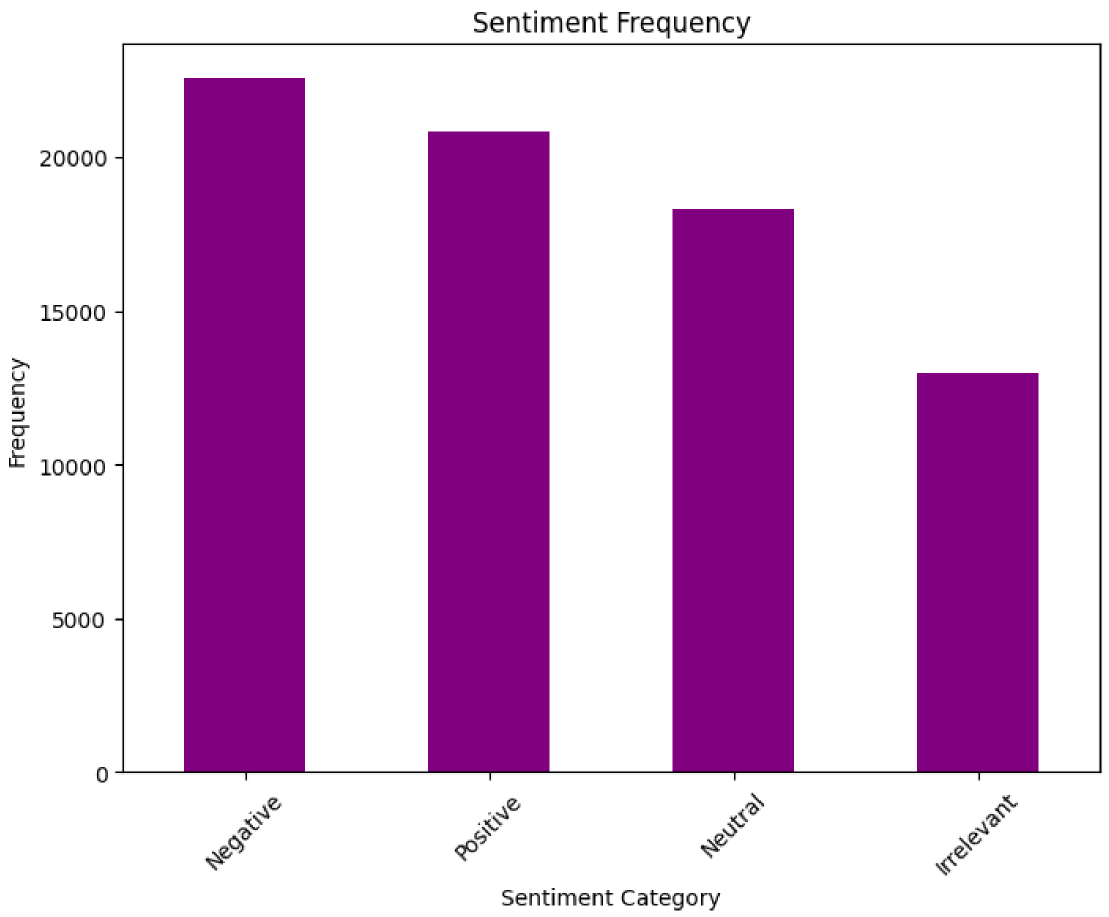
```
[ ]: #Plot sentiment distribution
import matplotlib.pyplot as plt

plt.hist(df['sentiment_score'], bins=20, color='skyblue', edgecolor='black')
plt.title('Sentiment Distribution')
plt.xlabel('Sentiment Score')
plt.ylabel('Frequency')
plt.show()
```



Bar Chart of Sentiment Frequency:

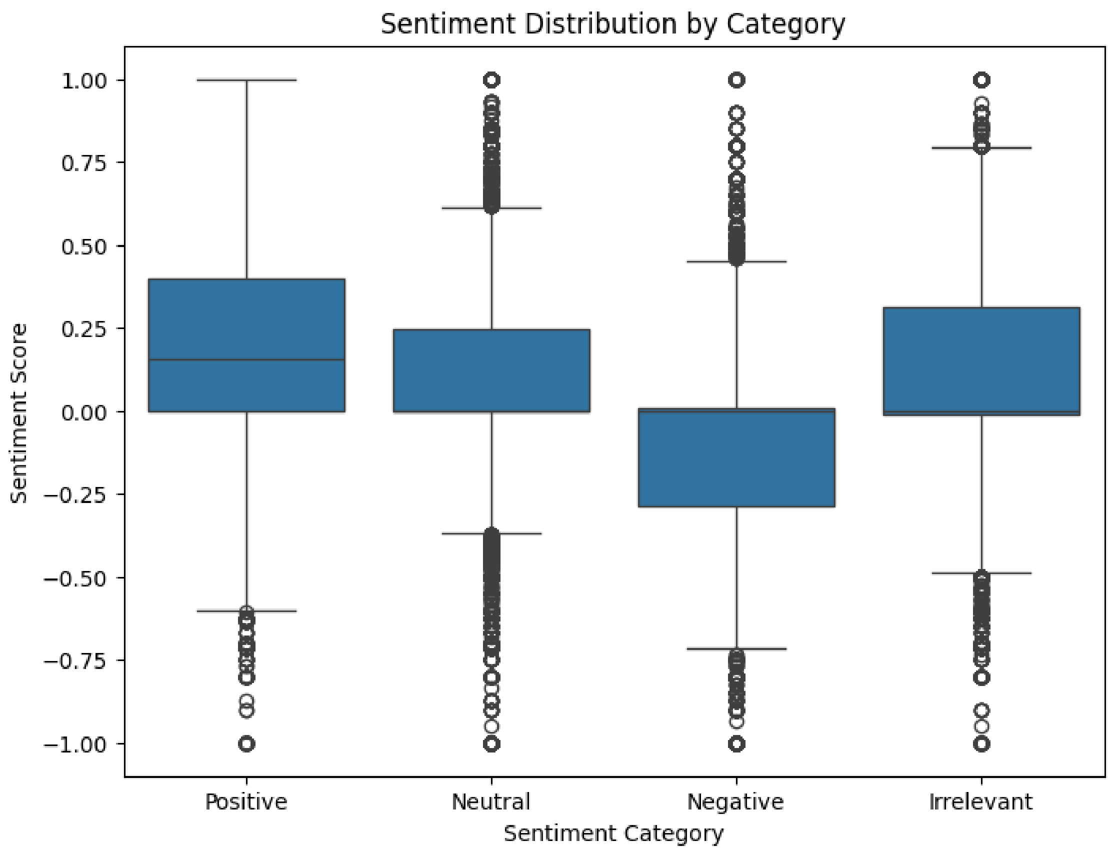
```
[ ]: #Plot a bar chart to show the frequency of each sentiment category.
     plt.figure(figsize=(8, 6))
     df['labels'].value_counts().plot(kind='bar', color='purple')
     plt.title('Sentiment Frequency')
     plt.xlabel('Sentiment Category')
     plt.ylabel('Frequency')
     plt.xticks(rotation=45)
     plt.show()
```



Boxplot by Sentiment Category:

Visualize the distribution of sentiment scores for different categories (e.g., positive, negative, neutral).

```
[ ]: import seaborn as sns
     plt.figure(figsize=(8, 6))
     sns.boxplot(x='labels', y='sentiment_score', data=df)
     plt.title('Sentiment Distribution by Category')
     plt.xlabel('Sentiment Category')
     plt.ylabel('Sentiment Score')
     plt.show()
```



[ ]:
```
from nltk.sentiment import SentimentIntensityAnalyzer
nltk.download('vader_lexicon')
```

```
[nltk_data] Downloading package vader_lexicon to /root/nltk_data…
```

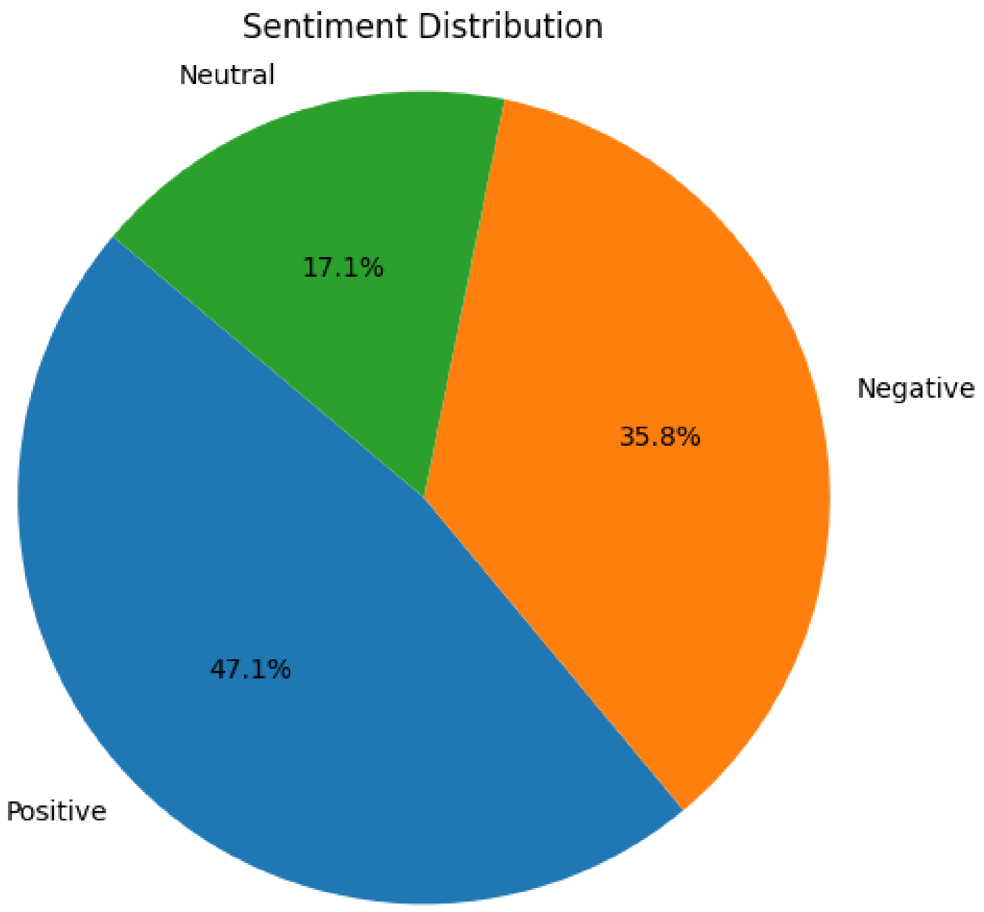
[ ]:
```
True
```

** Pie chart for sentiment distribution**

[ ]:
```
# Sentiment analysis
sid = SentimentIntensityAnalyzer()
df['sentiment'] = df['clean_text'].apply(lambda x: sid.
 ↪polarity_scores(x)['compound'])

# Visualization - Pie chart for sentiment distribution
sentiment_counts = df['sentiment'].apply(lambda x: 'Positive' if x > 0 else␣
 ↪('Negative' if x < 0 else 'Neutral')).value_counts()
```

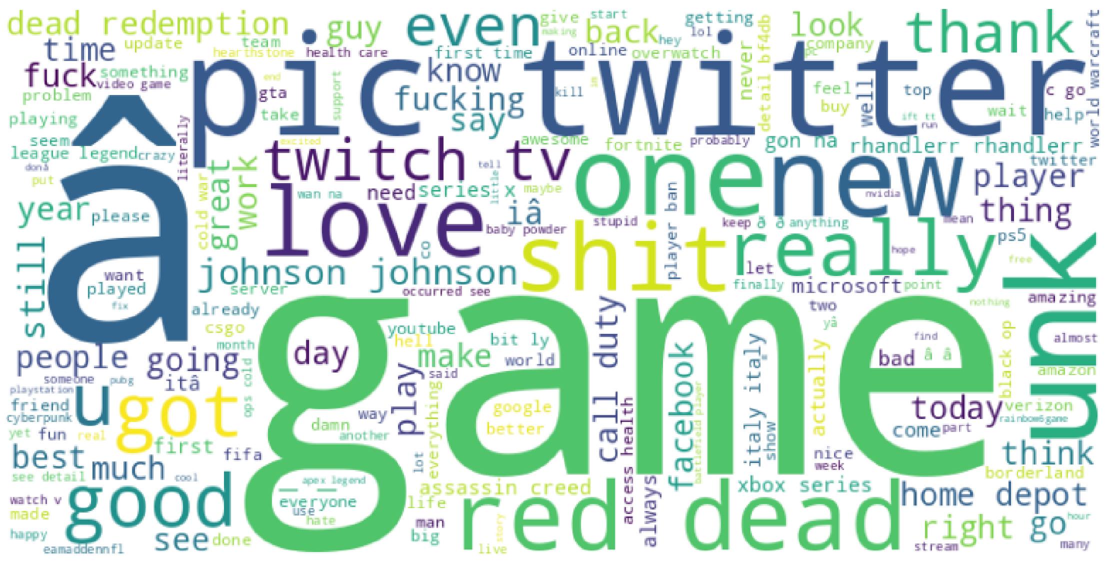
```
plt.figure(figsize=(6, 6))
plt.pie(sentiment_counts, labels=sentiment_counts.index, autopct='%1.1f%%',␣
↪startangle=140)
plt.title('Sentiment Distribution')
plt.axis('equal')
plt.show()
```



Word cloud

```
[ ]: # Visualization - Word cloud
text = ' '.join(df['clean_text'])
wordcloud = WordCloud(width=800, height=400, background_color='white').
↪generate(text)
plt.figure(figsize=(10, 6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.title('Word Cloud')
plt.axis('off')
plt.show()
```

Word Cloud


game
pic twitter
â
one new
love
shit
really
unk
red dead
good
got
twitch tv
even
thank
dead redemption
johnson johnson
call duty
facebook
home depot
people
best
still